Date=25/08/2020
Lecture By=Shubham Joshi
Subject ⇒ DP=3

IN PREVIOUS LECTURE **(QUICK RECAP) Date-24/08/2020**	**In Today's Lecture (Overview)**
Question ⇒ Maximum profit from sale of wines Mcqs Questions For Self Practice // Assignment And CC for the Day	Question=1 Find the longest Increasing subsequence Question=2 Find the Longest Increasing SubArray MCQs Questions for Self Practice // CC for the Day

In Today's Lecture
We discussed some problems regarding Dynamic Programing 1.find the longest increasing subsequence 2.longest increasing Subarray

Question=1 Find the longest Increasing subsequence

The Longest Increasing Subsequence (LIS) problem as an example problem that can be solved using Dynamic Programming.

The Longest Increasing Subsequence (LIS) problem is to find the length of the longest subsequence of a given sequence such that all elements of the subsequence are sorted in increasing order. For example, the length of LIS for {10, 22, 9, 33, 21, 50, 41, 60, 80} is 6 and LIS is {10, 22, 33, 50, 60, 80}.

arr[ ]	10	22	9	33	21	50	41	60	80
LIS	1	2		3		4		5	6

Examples:

Input: arr[] = {3, 10, 2, 1, 20}
Output: Length of LIS = 3
The longest increasing subsequence is 3, 10, 20

Input: arr[] = {3, 2}
Output: Length of LIS = 1
The longest increasing subsequences are {3} and {2}

Input: arr[] = {50, 3, 10, 7, 40, 80}
Output: Length of LIS = 4
The longest increasing subsequence is {3, 7, 40, 80}

Code

```
# Dynamic programming Python implementation
# of LIS problem

# lis returns length of the longest
# increasing subsequence in arr of size n
def lis(arr):
    n = len(arr)

    # Declare the list (array) for LIS and
    # initialize LIS values for all indexes
    lis = [1]*n

    # Compute optimized LIS values in bottom up manner
    for i in range (1 , n):
        for j in range(0 , i):
            if arr[i] > arr[j] and lis[i]< lis[j] + 1 :
                lis[i] = lis[j]+1

    # Initialize maximum to 0 to get
```

```
    # the maximum of all LIS
    maximum = 0

    # Pick maximum of all LIS values
    for i in range(n):
        maximum = max(maximum , lis[i])

    return maximum
# end of lis function

# Driver program to test above function
arr = [10, 22, 9, 33, 21, 50, 41, 60]
print ("Length of lis is", lis(arr))
```

Question=2 Find the Longest Increasing SubArray

Note=this Question's Approach Was only discussed in the lecture but the problem itself was not explained so i am just discussing the Approach for the same

Given an array containing **n** numbers. The problem is to find the length of the longest contiguous subarray such that every element in the subarray is strictly greater than its previous element in the same subarray. Time Complexity should be O(n).

Examples:

```
Input : arr[] = {5, 6, 3, 5, 7, 8, 9, 1, 2}
Output : 5
The subarray is {3, 5, 7, 8, 9}

Input : arr[] = {12, 13, 1, 5, 4, 7, 8, 10, 10, 11}
Output : 4
The subarray is {4, 7, 8, 10}
```

Algorithm:

```
lenOfLongIncSubArr(arr, n)
    Declare max = 1, len = 1
    for i = 1 to n-1
    if arr[i] > arr[i-1]
        len++
    else
        if max < len
            max = len
        len = 1
    if max < len
        max = len
    return max
```

MCQs

1.**What is the time complexity of longest increasing subsequence dp solution?**

A=O(2^n)

B=O(n)

C=O(n2)

2.**What is the space complexity of longest increasing subsequence problem ?**

A=O(n)

B=O(1)

C=O(n2)

3.**Which of these is a subsequence in array ? [1 2 3 4 5]**

A=4 3 5

B=2 4 5

C=4 1 5

4.**Which of these is a subarray for array [1,2,3,4,5]**

A=[3,4,5]

B=[1,3,4]

C=[1,2,5,4]

Questions for Self Practice // CC for the Day

https://leetcode.com/problems/longest-increasing-subsequence/